METHOD TO PROXY IP SERVICES

[0001] This application relates to U.S. provisional application 60/274,209 filed March 9, 2001.

Field of the Invention:

[0002] The present invention relates to a method to proxy IP services on devices that are located within networks that have non-routable private addresses.

Background to the Invention:

[0003] With the proliferation of TCP/IP technology worldwide, including outside the Internet itself, an increasing number of enterprises have used private Internet addresses for intra-enterprise communications, without any intention to ever directly connect to other enterprises or the Internet itself. Such addresses are not globally unique, and often not even organizationally unique. Such networks use Network Address Translation (NAT) to communicate with devices outside their domain.

[0004] Network Address Translation (NAT) is a known method by which IP addresses are mapped from one realm to another, in an attempt to provide transparent routing to hosts. Traditionally, NAT devices are used to connect an isolated address realm with private unregistered addresses to an external realm with globally unique registered addresses. In a typical NAT configuration a single externally visible IP host acts as a transparent gateway to the private Internet addresses with a network. The devices in the private network appear to have the same IP address to devices outside the domain. There is no way to discriminate between them. This is called one-to-many NAT. Such a scheme has allowed rapid deployment of enterprise TCP/IP

networks as it permits enterprises to have extreme flexibility with the number of IP addresses that they can use internally while still having transparent access to Internet services.

[0005] A problem exists when dealing with multiple domains of private addresses, as they are not globally unique. A single enterprise may have several departments that each uses the same private addressing scheme. An external vendor may have several clients that have numbering that is organizationally unique, but has conflict with the addressing in other organizations. This is a common problem, as there are only three sets of private Internet addresses. The Internet Assigned Numbers Authority (IANA) has reserved the following three blocks of the IP address space for private Internets:

10.0.0.0 - 10.255.255.255 (10/8 prefix)

172.16.0.0 - 172.31.255.255 (172.16/12 prefix)

192.168.0.0 - 192.168.225.225 (192.168/16 prefix)

[0006] Note that the first block is nothing but a single class A network number, while the second block is a set of 16 contiguous class B network numbers, and the third block is a set of 256 contiguous class C network numbers. An enterprise that decides to use IP addresses out of the address space defined in this document can do so without any coordination with IANA or an Internet registry. The address space can thus be used by many enterprises. This has created a situation where there is massive addressing conflict between networks.

[0007] TCP/IP routing requires that all hosts in the routed domain be unique. There cannot be any conflicts. In networks where there are private address ranges the networks must be isolated via methods such as one-to-

many NAT. Such devices will be able to create sessions with devices on other networks that have globally unique addresses. However, an outside device will see it as a connection from the masquerading host, not the actual device. Furthermore, devices outside these networks cannot create sessions with devices inside these networks using the actual IP address of the devices in question, as the one-to-many relationship only works one way and traditional IP routing has no solution for accessing private networks from the public network and cannot operate at all if these are IP address conflicts.

[0008] There is no need for methods that allow access to devices in private networks from the public network. There is also a need for methods that uniquely identify devices that have private IP addresses even when these addresses are in conflict with those in other networks. The methods have to take into account a variety of network topologies and path routes between a client and a device with which it wises to communicate.

Identification of Devices

[0009] A network management system discovers devices and their attributes. Apart from an IP address, devices may have Media Access Control (MAC) addresses, unique and local DNS names, SNMP system names, Windows names and several other discriminators. The user can select a device uniquely using one of a choice of metrics. The number of possible discriminators is unbounded and changing. New metrics, such as Voice over IP telephone number, are appearing as new products appear.

[00010] A network management system determines the physical topology of one or more networks. Determining the physical topology of the network allows a master proxy to determine that more than one device in its list

has the same IP address and be able to discriminate between them. This is possible if an only if the topology is not referenced by IP address but by a different discriminator. In systems that use IP address as database key such discrimination is impossible. The method in U.S. Patent 5,926,462 issued July 20, 1999 to Schenkel et al could be used to create a topology database that allows such discrimination.

Firewall Rules

[00011] A network may have a set of firewall rules that cannot be obtained by a network management system. An additional data source describing this information will be needed. A device inventory with attributes and connectivity information in conjunction with the rules needed to access firewalls in the network completes the seeding of proxies.

Summary of the Invention:

[00012] The present invention uses a network management system to identify and place devices. HTTP redirection and proxy servers are used to select and access devices that have IP address range conflicts with other devices, and in non-routable private networks, or behind network firewalls. A master proxy then determines which proxies, if any, are used to communicate with a specific device. A user accesses the service via an HTTP compliant client. The primary proxy then redirects the client to the appropriate device, be it the device itself or a proxy for the device. The URL of the request contains within itself a message that allows the proxy to find out which device is being acted upon and what protocol action to take. Like HTTP itself the protocol is connectionless. Each request requires a

unique HTTP session. The method is compliant with HTTP protocols 0.9, 1.0 and 1.1.

[00013] In accordance with an embodiment of the invention, a method for providing a proxy service in a computer network, is comprised of the steps of: receiving a request to access a device, determining the path to the device, ascertaining what firewall rules exist for that given path, and redirecting the client to the appropriate proxy, if any is needed, for that path.

Selection of Paths

[00014] The method of the present invention allows for four proxy methods for a given device.

1. A proxy server identifies the device and the client can access the device directly.
2. A proxy server can identify and access the device but it is inaccessible to the client.
3. A proxy server can identify the device but access is through a second proxy server. The second proxy server is accessible to the client.
4. A proxy server can identify the device but access is through a second proxy server. The second proxy server is inaccessible to the client.

[00015] The methods are recursive. Methods 3 and 4 are recursions of 1 and 2, and the methods can be joined and extended indefinitely. Once a proxy is seeded it can determine which path to take to make a proxy connection between a client and a device.

HTTP Redirection

[00016] The invention redirects clients to the device or proxy by using an HTTP redirect message which informs the client of the address to which to redirect itself.

Transparent Proxies

[00017] Each proxy acts transparently and cumulatively. No client-side configuration for the proxy is needed.

Authentication

[00018] The master proxy server has an authentication and access control method for the client. Authentication between proxies is transparent to the user. Such authentication can be either in-band, via cookies or basic HTTP authentication, or out of band, by access control lists or database lookups.

Connectionless Protocol

[00019] HTTP is a connectionless protocol, each request is an independent session. In HTTP protocol versions 0.9 and 1.0, once a document is transmitted the TCP session closes. However, HTTP 1.1 allows for a TCP socket to remain open after the request has been made. The invention allows for maximum flexibility in determining which, if any TCP sessions remain open.

Brief Description of the Drawings:

[00020] A person understanding the above-described invention may now conceive of alternative designs, using the principles described herein. All such designs which fall within the scope of the claims appended hereto are considered to be part of the present invention.

[00021] Figure 1 is a block diagram of a circuit for configuring proxies;

[00022] Figure 2 is a block diagram of a proxy server redirecting to an HTTP server;

[00023] Figure 3 is a block diagram of a proxy server forwarding to an HTTP server;

[00024] Figure 4 is a block diagram of a proxy server redirecting via a second proxy server to an HTTP server; and

[00025] Figure 5 is a block diagram of a proxy session through multiple proxy servers to an HTTP server.

Detailed Description of the Invention;

[00026] Referring to Figure 1, there is shown a block diagram of a system for configuring proxy servers, hereinafter proxies. The lower portion of the drawing graphically shows the state transitions of the system of Figure 1. A network management system (NMS) 10 is connected to a communications network 11 and to a database store 12. Initially the NMS10 discovers devices and their attributes, which is illustrated graphically at A between 10 and 11 and as step A in the state transitions. Next the NMS 10 stores devices attributes and their connectivity in the database 12, as shown at B in the drawings. The proxy configuration 13 is seeded device and attribute information as well as device location at C. Firewall information from Firewall Rules 14 is fed to the proxy configuration 13 at step D. The supplying of firewall information may either be manual or automatic. Proxy paths 15 between device pairs are determined and stored at step E. Proxies 16 then obtain the path list from proxy paths 15 at step F and are configured.

[00027] In Figure 2, a proxy server 20 identifies the device 21 and the client 22 can access the device 21 directly. Step A is further subdivided into A_s, an HTTP Authorize/Redirect Start step and A_S, an HTTP Authorize/Redirect Finish step, which are shown on the Figure 2 state transition diagrams. Step B is also subdivided into B_s an HTTP Request/Response Start, and B_F an HTTP Request/Response Finish step also shown on the state transitions diagram.

[00028] In Figure 3, a proxy 30 forwards to an HTTP server, when the client 31 seeks a connection to device 32. As in Figure 2, A_S, A_F, B_S and B_F indicate the same steps in the state transitions, while C_S indicates an HTTP Proxy Request/Response start, and C_F indicates a Proxy Request/Response Finish. In this case a proxy server 30 can identify and access to the device 32 but the device 32 is inaccessible to the client 31.

[00029] In Figure 4, a client 40 accesses the proxy 41 which redirects to a second proxy 42 which is accessible to the client 42,and proxy 42 is accessible to the client 40. The state transitions are shown wherein A_S, A_F, B_S, B_F, C_S and C_F are as defined in relation to Figure 3, and D_S indicates an HTTP proxy Request/Response start and D_F indicates an HTTP proxy Request/Response finish. As before the arrows in the State Transitions are indicative of the steps in the connection process, the oval arrow indicating a recursive step, such as B_F to B_S in Figure 3, and C_F to C_S in Figure 4. In this example shown in Figure 4, the proxy 41 can identify the device 43, but access is through proxy 42, and proxy 42 is accessible to client 40.

[00030] A further example is shown in Figure 5 in which access is obtained through multiple proxies to an HTTP server. As before, a client 50 accesses a proxy 51 at A which can identify the device 53, but access is through a second proxy 52 at B and the second proxy 52 is inaccessible to the client 50. The state transitions A_S, A_F, B_S, B_F, C_G, C_F, D_S, D_F are as explained in relation to Figure 4, and E_S is an HTTP proxy Request/Response start, and E_F is a proxy Request/Response finish. The recursive portion of the transitions is shown by the

elliptical arrow, with the letters, A, B, C, D and E illustrating the states of the process from client 50 to proxy 51 to proxy 52 to device 53, and back through proxy 52 to proxy 51 and to client 50.

Other Applications of the Invention

[00031] The invention may also be used to proxy any connection-oriented TCP service. Typical services that can be supported by the invention include telnet and ftp. The invention can be used to launch any tcp service that can be launched using a url within a browser. The example below is for an application of this invention for the telnet protocol.

[00032] Launching of a telnet or ftp client is compliant with HTTP protocols 0.9, 1.0 and 1.1.

Proxy Configuration

[00033] Proxy configuration is identical to the method used for http servers.

Telnet URL

[00034] The invention redirects clients to the device or proxy by using a telnet url which will launch a telnet client that instantiates a connection using the ip address and TCP port specified in the URL. The URL is formatted as follows:

telnet://{ip}:{tcp port}

where 'telnet' is the protocol specifier. {ip} is either numeric IP address or fully qualified domain name, and {tcp port} is the tcp port that is used for the connection.

FTP URL

[00035] The invention redirects clients to the device or proxy by using a ftp url which will launch an ftp client that instantiates a connection using the ip

address and TCP port specified in the URL. The URL is formatted as follows:

ftp://{ip}:{tcp port}

where ftp is the protocol specifier, {ip} is either a numeric IP address or fully qualified domain name, and {tcp port} is the tcp port that is used for the connection.

[00036] A person understanding the above-described invention may now conceive of alternative designs, using the principles described herein. All such designs which fall within the scope of the claims appended hereto are considered to be part of the present invention.